

# Learning-based Floor Segmentation and Reconstruction

Jose Pardeiro, Javier V. Gómez, David Álvarez and Luis Moreno

Robotics Lab, Department of Systems and Automation, University Carlos III of Madrid, Leganés, 28911, Spain  
jose.pardeiro@gmail.com, {jvgomez, dasanche, moreno}@ing.uc3m.es  
WWW home page: <http://roboticslab.uc3m.es/>

**Abstract.** This paper presents a comparison of the different colour spaces used in an environment modelling algorithm. This algorithm is based on the fusion of depth and colour information of a low-cost RGB-D camera to model an indoor environment. This modelling is based on creating and updating Gaussian models of the colour of the walkable floor. The analysis carried out tests the performance of three different colour spaces, obtaining the best choice to get a correct floor segmentation and reconstruction. As the results show, the algorithm performance highly depends on the colour space chosen. The method has been evaluated in a set of frames representing different environments captured with a RGB-D camera.

**Keywords:** Floor segmentation, colour spaces, RGB-D sensor

## 1 INTRODUCTION

Nowadays a correct environment modelling is a very important technique to develop good mobile robots. To achieve a reliable autonomous motion the robot has to know which zones are walkable and which ones are occupied by obstacles. These obstacles can be static, such as paper bins, boxes or furniture, or dynamic like people walking, and the robot has to be able to separate these obstacles from the rest of the environment to detect the walkable zones.

Traditionally researchers have used different sensors trying to generate a tridimensional representation of the environment with different algorithms and techniques. In these techniques, the main purpose consists on detecting the free space immediately around the mobile robot rather than the specific wall-floor boundary. Some examples can be found in [1], where 4 stereo cameras are used to accomplish this task. An alternate approach was proposed in [2], using a combination of colour and gradient histograms to distinguish free space from obstacles.

Only a few of researchers have considered the floor segmentation a problem by itself. The techniques employed by these researchers want to utilise the ground plane restriction in different forms like planar holographs to optical flow vectors [3, 4].

In 2005 the Department of Defense from USA organized a contest called DARPA Grand Challenge, consisting in moving a car from one point to another one without any human intervention. The Stanford University won this contest. Among the different systems implemented in the car, they used a segmentation of walkable floor algorithm which allowed their car to circulate at the highest average velocity [5]. For this task they used two different sensors: laser range finders and a standard RGB camera. Their solution combined the information of the sensors to be able to increase the walkable floor segmentation capability.

Our algorithm is based on the same idea. However, we focus on indoor environments and a low-cost RGB-D camera is used. This type of camera provides a modern technology at low cost, so it is not necessary to have a very expensive equipment to get a correct modelling. However, some researches about the performance of a RGB-D camera in environment modelling has been done with a successful results [6–8]. In this work, some common problems of image processing in indoor environments such as: brightness and colour changes caused by artificial lightning, similarity among different areas in the environment and so on, are addressed, and a measure of its performance is presented.

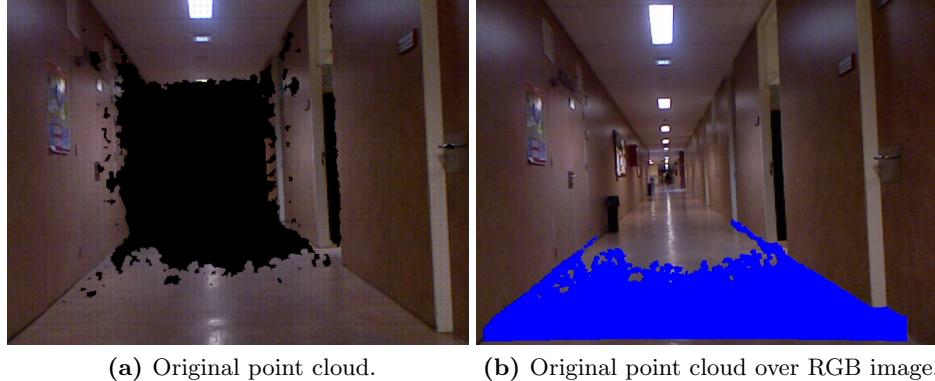
The next sections of the paper are organised as follows. In section 2 the description of the algorithm is done. Section 3 contains the experimental results and their analysis. Finally, in section 4 conclusions and future work are addressed.

## 2 SEGMENTATION OF WALKABLE FLOORS

In this algorithm a low-cost RGB-D camera has been used. These cameras consist of a standard RGB camera and a set of depth sensors able to measure distances. These sensors are capable of detecting the environment geometry in a range of about 7 meters, but this range may not be enough depending on the application and the velocity of the robot. For example, in figure 1 a capture of an RGB-D camera located in the middle of a corridor is shown. As it can be seen, the range measured by the RGB-D camera depth sensors is lower than the RGB image range. One of the main goals of this algorithm is to overcome this limitation merging the information provided by both sensors. The algorithm used for the fusion of this information is based on the research presented in [5].

The employed algorithm is based on learning the colour parameters of the floor segmented with the depth information. The estimation of the walkable floor out of range of the depth sensors is based on the assumption that the floor has the same colour properties than the previously segmented floor. Using this supposition the algorithm is able to estimate the walkable zones located outside of range of the depth sensors.

The algorithm starts working with the point cloud obtained by the RGB-D camera depth sensors. With this information a tridimensional representation of the environment can be obtained. First, these data is processed to be able to detect and segment the floor. Later, a transformation from points in the 3D space to pixels in the RGB image is needed to get colour parameters of the floor.



**Fig. 1:** Example of image captured with RGB-D camera.

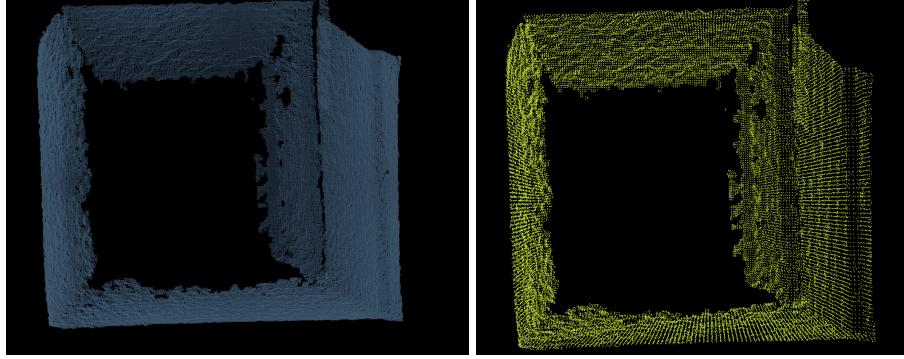
To reduce the high quantity of information, a clustering technique is used to group it and the resulting groups are defined by a simple statistical function to reduce the time consumption of the next processes. Finally, a statistical measure of the distance between each pixel and the functions obtained before is used to classify the colour pixels. When the colour of a pixel is considered to be near the function that defines the colour parameters of the floor, this pixel is marked as part of the floor. In the opposite case, the pixel is discriminated and considered as an obstacle. All these steps are detailed in the following points.

## 2.1 Filtering

The algorithm begins with a filtering process. The filtering consists on reducing the amount of data obtained as much as possible while maintaining the desired features of the input. With this process the algorithm improves its speed and avoids some future problems due to the excess of information. The filter used is a voxel grid filter.

The voxel grid filter generates a grid of cubes which have an identical size. This filter is applied to all the data of the cloud. All the points which belong to the same voxel in the grid are replaced by their centroid. The output data keeps the original geometrical information of the environment but has fewer points and a more uniform density.

The data reduction obtained depends on the cube size, so the output grid obtained is bigger when the cube is smaller. Finding an appropriate voxel grid size with the objective to achieve an equilibrium between speed and filter quality is a critical, task dependent problem. An aggressive filtering can erase relevant information but a low filtering makes the algorithm to be excessively slow producing errors in the following steps. The decision on the size of the cube has been made empirically and the result shown in figure 2.



(a) Original point cloud with 208974 points. (b) Filtered cloud with cube size 3 cm with 36242 points.

**Fig. 2:** Input cloud filtering.

## 2.2 Segmentation of the point cloud

The second step of the algorithm consists on the segmentation of the point cloud into the different main planes that belong to the cloud. To compute these planes the RANSAC [9], abbreviation of RA<sup>N</sup>dom SA<sup>M</sup>ple C<sup>O</sup>nensus, algorithm is used. This method adjusts, in an iterative way, the parameters of the plane that contain the highest quantity of points. To minimise possible errors, RANSAC also evaluates the distance between different points, discriminating the points that overcome a limit.

Mathematically a plane can be described as:

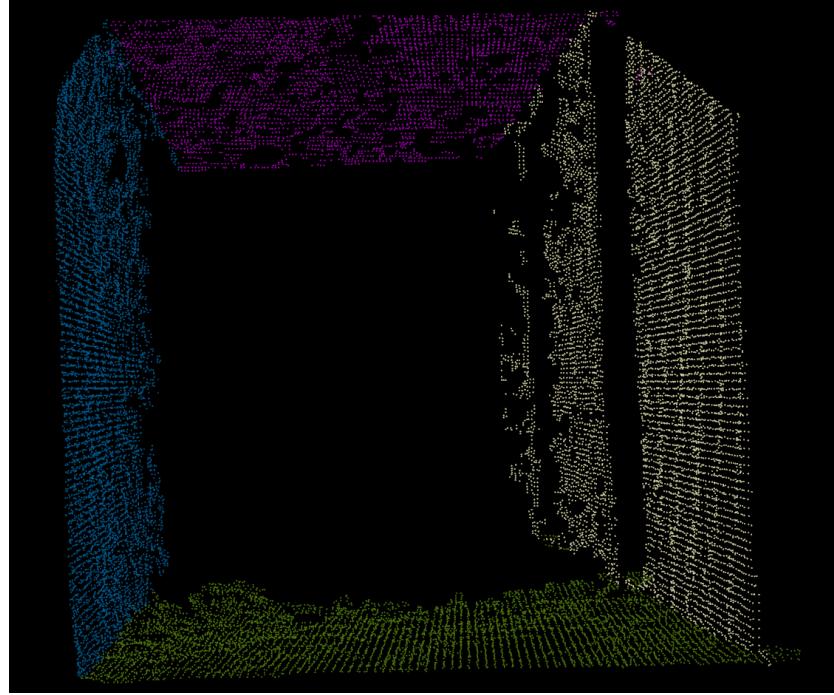
$$ax + by + cz + d = 0 \quad (1)$$

When the coefficients of all the planes are obtained, they are analysed to determine their position in the 3D environment. For this purpose, the camera is assumed to be placed so that the z axis of the depth sensor is parallel to the floor. Using the camera as a reference, a plane is considered to be horizontal if  $a, c \approx 0$  and  $b \approx \pm 1$  since  $y$  is constant for every  $(x, z)$ , and the height of the plane is  $y \approx -d/b$ . Therefore, if  $y < 0$  the plane is labelled as floor and as ceiling otherwise. The results of the segmentation of the planes are shown in figure 3 and table 1.

## 2.3 Pixels which pertain to the floor

Once the planes have been analysed and the floor identified, the colour information of the floor is extracted to be analysed. This is done using the pin-hole model described on equation 2.

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_i/Z_i \\ Y_i/Z_i \\ 1 \end{pmatrix} \quad (2)$$



**Fig. 3:** Point clouds segmented.

**Table 1:** Coefficients of generated planes.

	a	b	c	d
Right wall	0.990	0.995	0.124	1.134
Left wall	0.995	-0.030	0.093	-1.204
Roof	-0.047	0.999	0.025	0.920
Floor	-0.044	-0.999	-0.01	-1.954

where  $u$  and  $v$  represent the pixel position,  $f_x$  and  $f_y$  the focal distance on  $x$  and  $y$  axes expressed in pixel-related units,  $C_x$  and  $C_y$  the principal point coordinates and  $X_i$ ,  $Y_i$  and  $Z_i$  represent the real distances on the three Cartesian axes.

In the previous step the floor plane was obtained. Applying the formula described on 2 to the points belonging to the floor the corresponding pixel in the RGB image  $(u, v)$  can be obtained. From these pixels the colour information is extracted, and the result is shown on the figure 4



**Fig. 4:** Example of floor points projected as pixels.

## 2.4 Colour parameters analysis

In the original algorithm [5], all the computation needed to compare the colour parameters is simplified due to the reduction of the information of the pixels to statistical functions. In our algorithm, among all the possibilities for clustering, the k-means [10] technique has been selected. This technique divides the input data into  $k$  different groups based on a distance function. When the clustering has finished, the obtained groups are reduced to 3-dimensional Gaussian distributions, where each dimension is a parameter of the colour space chosen. A Gaussian function can be defined using its mean and its covariance. To calculate these parameters, equations 3 and 4 are used.

$$\mu = \frac{1}{n} \sum_{i=1}^n X_i \quad (3)$$

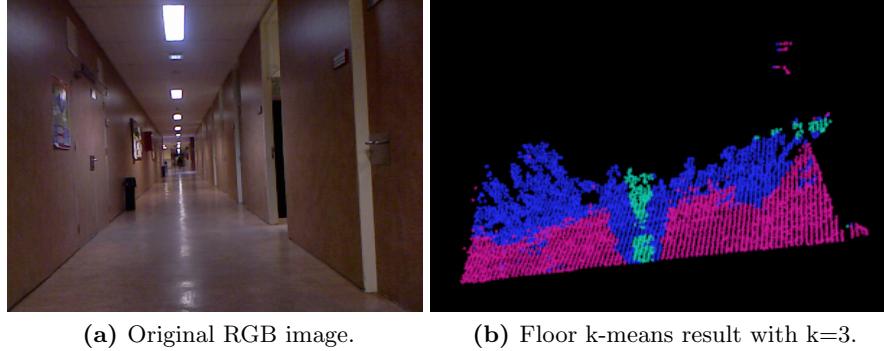
$$\sigma^2 = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 \quad (4)$$

The results of this process are shown in figure 5 and table 2. Each Gaussian function is assigned a mass which represent the number of points in the input

floor which are included in that Gaussian. This is useful as a measure of the importance of each Gaussian.

## 2.5 Region of interest

Since the labelling of the pixels is colour-based, it could happen that some zones of the image are labelled as floor when they actually belong to walls, ceiling or obstacles. Thus, the introduction of a region of interest (ROI) when analysing the colour information is required. This region will delimit the area in which it is more likely to find pixels which belong to the floor.

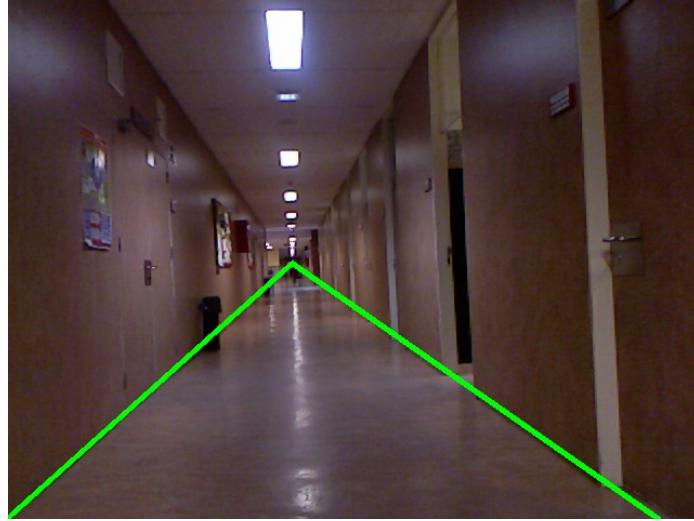


**Fig. 5:** Example of the Gaussian distributions clustering.

**Table 2:** Example of the Gaussian representation combined with K-means clustering.

	R	G	B
Means 1	122.773	109.447	122.984
Variances 1	249.702	278.566	262.49
Mass 1		309	
Means 2	81.9109	62.0661	62.4672
Variances 2	30.2974	31.2947	77.3858
Mass 2		3221	
Means 3	94.288	78.1837	85.6717
Variances 3	35.3101	30.9852	60.289
Mass 3		309	

Considering simple corridors, the initial point cloud will be composed by 4 main planes: floor, walls and ceiling. As the parameters of the 3D planes have been extracted in previous steps, it is possible to find out the 3D line resulting of the intersection between the walls and the floor. Following, the computed lines are projected so the ROI parameters are expressed in terms of pixels, as shown in figure 6.



**Fig. 6:** Example of a region of interest.

## 2.6 Floor labelling

In this step all the pixels within the ROI are analysed in order to determine whether they belong to the floor or are obstacles. This labelling is done according to the distance between the Gaussian distributions of the learnt floors ( $\mu_{L,i}$ ,  $\Sigma_{L,i}$ ) and the colour of the pixel  $(u, v)$ , denoted as  $x_{(u,v)}$ . In this case, the metric employed is the Mahalanobis distance  $M_i(x_{(u,v)})$ :

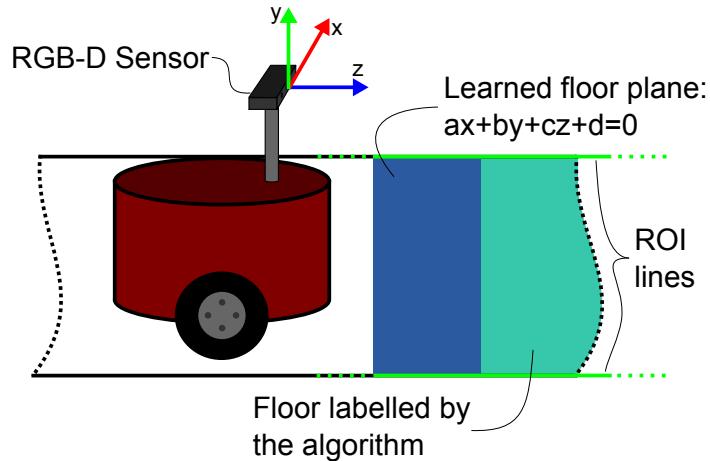
$$M_i(x_{(u,v)}) = \sqrt{(x_{(u,v)} - \mu_{L,i})^T \Sigma_{L,i}^{-1} (x_{(u,v)} - \mu_{L,i})} \quad (5)$$

Note that this model is independent of the colour space used. Eq. 5 is applied for every Gaussian distribution  $i$  obtained in section 2.4. If the minimum value  $M_i(x_{(u,v)})$  is below a given threshold  $th$ , the corresponding pixel  $x_{(u,v)}$  is labelled as floor  $i$ .

## 2.7 Map Reconstruction

This last step consists on reconstructing a map from the previous labelled data. Usually it is not possible to compute the inverse projection of the pixels because the depth information cannot be extracted with one single frame. However, in this case the pixels labelled as floor are known to lie in a 3D plane in the space. Together with the pinhole camera model, it is possible to extract the 3D position of every pixel. The pinhole camera model is defined by the equation 2 on section 2.3.

Assuming that the axis  $y$  is perpendicular to the plane  $Oxz$ ,  $\forall(x, z) \rightarrow z = K$ , and that every point labelled as floor will lie in the plane  $ax + by + cz + d = 0$ , as detailed in figure 7. Therefore, it is possible to compute the inverse projection:



**Fig. 7:** Kinect coordinates.

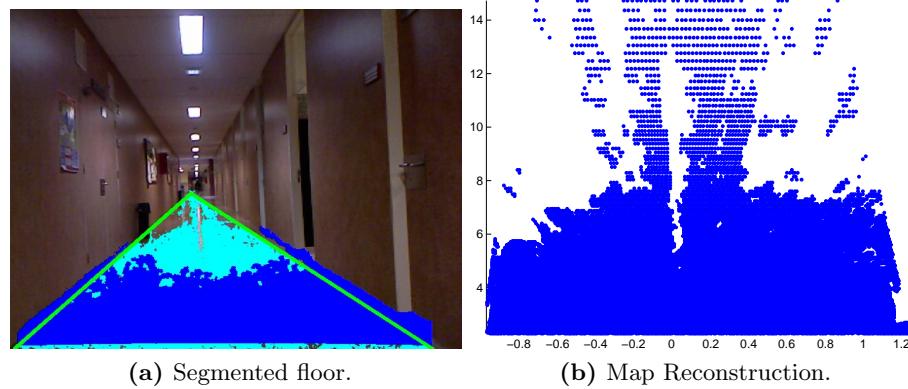
$$Y_i = -\frac{d}{b} \quad (6)$$

$$Z_i = f_y \frac{Y_i}{v - C_y} \quad (7)$$

$$X_i = \frac{(u - C_x)Z_i}{f_x} \quad (8)$$

The result is shown in the figure 8.

X



**Fig. 8:** Example of map generation.

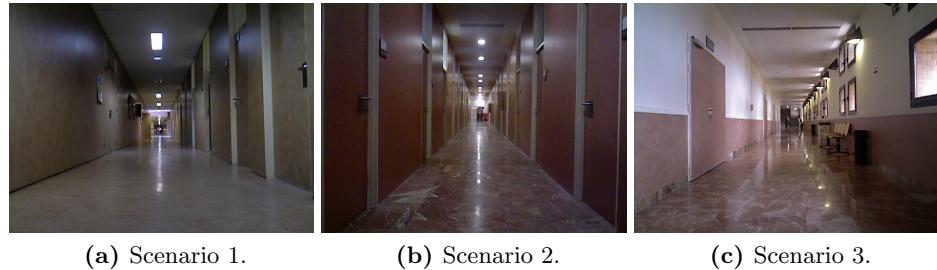
### 3 EXPERIMENTS

In this section a comparison among three different colour spaces is carried out: RGB [11], HSV and CIELAB [12]. Although the RGB-D cameras work with RGB colour space by default, the conversions among these spaces are well known [13].

The experiments have been carried out using C++ together with the PCL [14] and OpenCV [15] libraries.

#### 3.1 Experimental setup

Three different scenarios (corridors) have been chosen, shown in figure 9. Obstacles have been included in all of them but only two of them have been analysed with both obstacles and clear corridor. In each case 15 frames adquired from the same point of view have been analysed. 10 different values for  $th$  are given from 2 to 5 in RGB and LAB experiments and from 20 to 50 in HSV cases.



**Fig. 9:** Environments employed in the experiments.

The metrics employed for evaluation will be the Precision/Recall curves and also the F-score:

$$\text{Precision} = \frac{t_p}{t_p + f_p} \quad (9)$$

$$\text{Recall} = \frac{t_p}{t_p + f_n} \quad (10)$$

$$F = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (11)$$

where  $t_p$  stands for true positives,  $f_p$  for false positives and  $f_n$  means false negatives. The results are expressed in terms of means along all the frames for every value of  $th$ .

In this case, precision represents the rate of pixels labelled as floor correctly, among all the pixels actually considered as floor. Recall presents the sensitivity of the algorithm to detect floor pixels. And finally, the  $F$  term is a combination of the afore mentioned metrics evenly weighted.

### 3.2 Results analysis

First, the algorithm is applied to the two obstacles-free environments in order to test the its performance in an ideal case. As figure 10 shows, the precision is always 1 because all the pixels within the ROI are floor,  $f_p = 0$ . The recall improves since an increment in the  $th$  value makes the algorithm to be more permissive.

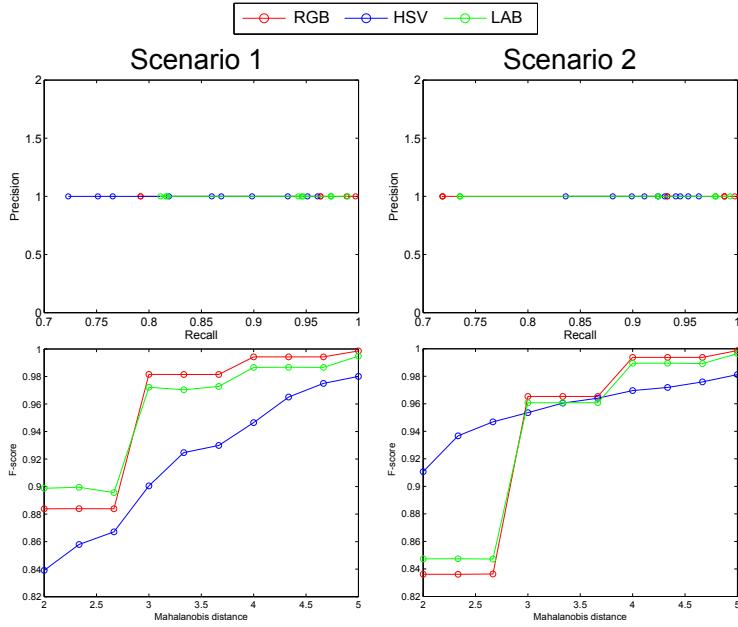
After, the algorithm is applied on three different environments with obstacles, such as people or paper bins, and the results are showed in figure 11.

In all cases, the precision of LAB and RGB are very similar, both of them higher than HSV. It means that HSV labels more obstacles as floor than the others. However, the HSV recall value is usually higher. As the  $th$  value becomes more permissive, the precision falls while the recall value increases. In other words, the algorithm reduces the number of  $f_n$  but the amount of  $f_p$  is higher.

Focusing on the F-score, for restrictive values of  $th$  the HSV colour space is more reliable. But if a balanced algorithm is desired, RGB and LAB are the better choices.

A visual example of the results is given in figure 12. In these images, dark blue represents the area labelled as floor by the depth sensors, and light blue represents the zone labelled as floor by the algorithm, all the images are evaluated with a permissive  $th$  value.

Besides, a 2D map representing the segmented floor is shown. In the latter, a good visual feedback of the different performance between the colour spaces is clearly seen. HSV example hardly avoids labelling the person in the picture as an obstacle, while the others present a good performance in close and far distances. However, all of them label the paper bin on the right as floor.



**Fig. 10:** Results in scenarios 1 and 2 with no obstacles.

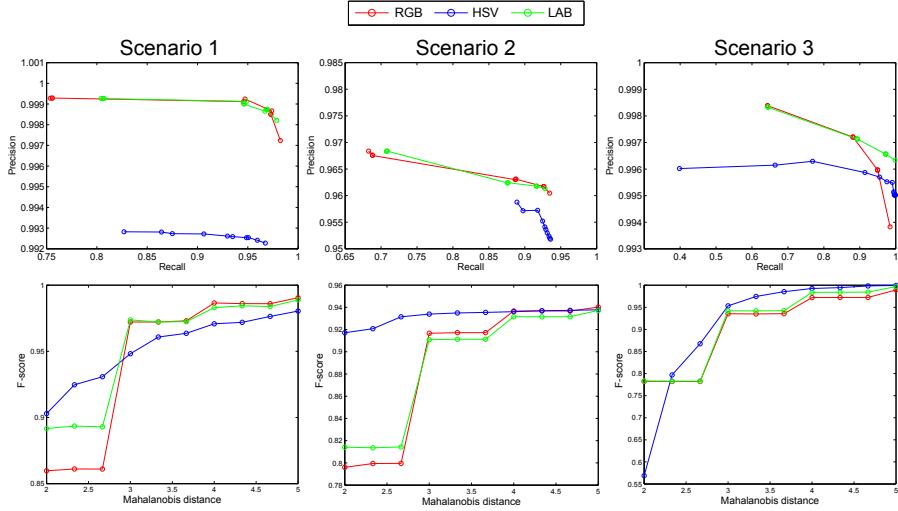
## 4 CONCLUSIONS AND FUTURE WORK

In this paper the performance of three different colour spaces has been analysed using an environment modelling algorithm. The algorithm has been applied on different environments, with and without obstacles, and Precision vs Recall and F-score graphics has been used to evaluate the performance.

LAB and RGB colour spaces have produced the best results on the algorithm with a moderate  $th$  value obtaining an equilibrium between floor labelled and obstacles labelled. HSV colour space is too sensible on the definition of a colour. For this reason this colour space obtains the best results when the  $th$  value is restrictive.

Some problems have been produced on different environments, which are a bit difficult to solve. When a bright area is in the depth sensor range, it usually considered as an independent Gaussian distribution, therefore this colour is labelled as floor and it will lead to errors. The same problem appears when an obstacle has the same colour than the floor, because the algorithm detects different pixels with a very similar colour. In these cases it is not possible to tell the difference between floor and obstacles by using only colour pixel information.

With this contribution a better colour space selection can be done. These results could be enough to choose the correct space colour in environment modelling applications based on colour data.



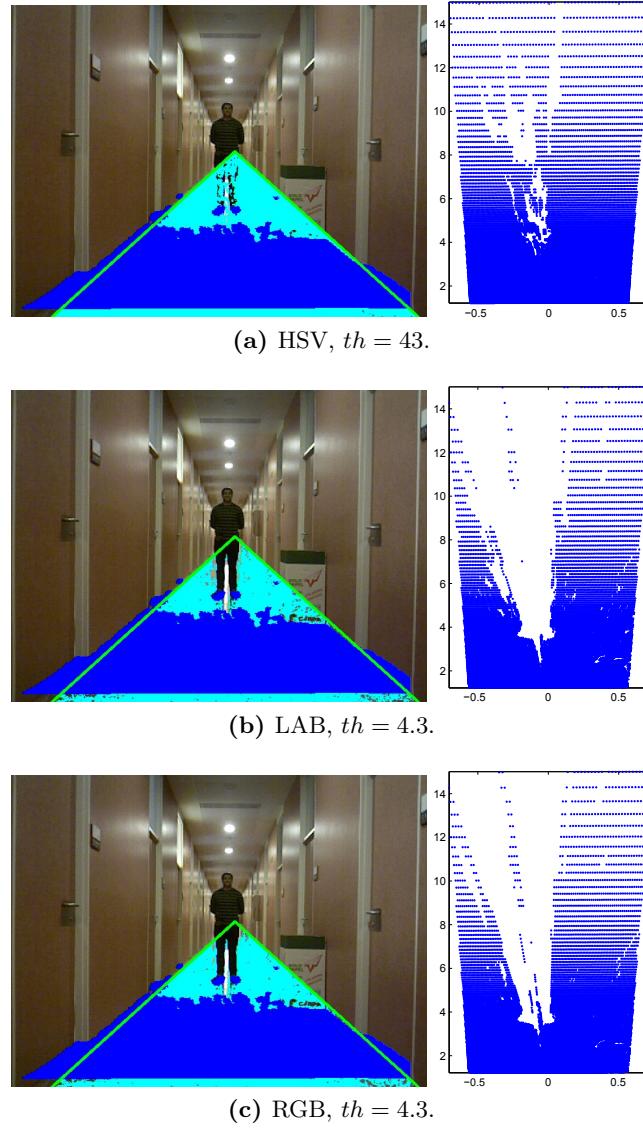
**Fig. 11:** Results in the three scenarios with obstacles in different positions. Mahalanobis values of HSV are multiplied by 10.

Future work is focused on improving the algorithm performance introducing new parameters related to the planes information to try to avoid some problems if the object has been detected by the depth sensor.

**Acknowledgement** This work was supported by the projects number DPI2010-17772 and CSD2009-00067 HYPER-CONSOLIDER INGENIO 2010 of the spanish MICYT.

## References

1. Sabe, K., Fukuchi, M., Gutmann, J.-S., Ohashi, T., Kawamoto, K., and Yoshigahara, T.: Obstacle avoidance and path planning for humanoid robots using stereo vision. Proceedings of the IEEE International Conference on Robotics and Automation (2004).
2. Lorigo, L. M., Brooks, R. A., Grimson, W. E. L.: Visually-guided obstacle avoidance in unstructured environments. IEEE Conference on Intelligent Robots and Systems. 1, 373-379 (1997).
3. Kim, Y.G., Kim, H.: Layered ground floor detection for vision-based mobile robot navigation. Proceedings of the IEEE International Conference on Robotics and Automation (2004).
4. Zhou, J. and Li., B.: Robust ground plane detection with normalised homography in monocular sequences from a robot platform. Proceedings of the International Conference on Image Processing (2006).
5. Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S., Bradski, G.: Self-supervised monocular road detection in desert terrain. Proceedings of Robotics: Science and Systems. Philadelphia, USA (2006).



**Fig. 12:** Example of the results for the environment 2. Left: colour space view. Right: 2D map reconstruction, scales are in meters.

6. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D Mapping: Using Depth Cameras for Dense 3D Modelling of Indoor Environments. International Symposium on Experimental Robotics (2010).
7. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D mapping: Using Kinect-style depth cameras for dense 3D modelling of indoor environments. The International Journal of Robotics Research. 1-17 (2012).

8. Pardeiro, J., Gómez, J. V., Álvarez, D., Moreno, L.: Estimación de suelos navegables para interiores. Robocity 2030: Robots sociales. Leganés, Spain (2013).
9. Fischler, M.A., Bolles, R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24. 381-395 (1981).
10. Kanungo, T.: An Efficient k-Means Clustering Algorithm: Analysis and Implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24. 7. 881-892 (2002).
11. Pascale, D.: A Review of RGB colour spaces...from xyY to R'G'B'. The Babel-Colour Company (2008).
12. Schanda, J.: COLORIMETRY: Understanding the CIE System. John Wiley and Sons, Inc. (2007).
13. Tkalcic, M., Tasic, J. F.: Colour spaces - perceptual, historical and applicational background. EUROCON 2003. Computer as a Tool. The IEEE Region 8.304 - 308 (2003).
14. Rusu R.B., Cousins S.: 3D is here: Point Cloud Library (PCL). Proc. Int. Conference on Robotics and Automation (ICRA) (2011).
15. Baggio, D. L., Emami, S., Millán Escrivá, D., Ievgen, K., Mahmood, N., Saragih, J., Shilkrot, R.: Mastering OpenCV with Practical Computer Vision Projects. Packt Publishing. Birmingham, UK (2012).